# Project Two README

## About the Project/Project Title

Training Dog Identification System

This software system, by Global Rain, has been developed to assist an innovative international rescue-animal training company, Grazioso Salvare, identify good candidates for search-and-rescue training by allowing Grazioso Salvare to examine records of the animals at five animal shelters in the region of Austin, Texas, USA.

## Motivation

This software provides several benefits to the Grazioso Salvare organization, since they will be able to access the records of their partners allowing them to have up to date records of all the current search and rescue candidates that will increase the efficiency of Grazioso Salvare’s dog training operation. The system will also allow for greater opportunities to stream-line Grazioso Salvare’s canine talent acquisition processes and provide an organized framework for the dog training operation to up-scale without losing track of prospective canine recruits.

## Getting Started

To utilize this software package a user will need to download a copy of the included CRUD python file, the Dashboard jupyter notebook document, and the assets folder that contains the Grazioso Salvare logo image and search-and-rescue dog image. After downloading these documents, ensure they have been saved into the same folder and then the user can run the Dashboard document within the jupyter notebook framework.

## Installation

There are 3 prerequisite software packages that are required to run this CRUD project, i.e. Python, MongoDB, and Jupyter Notebook. To install these programs follow the instructions listed below relating to your native OS.

In Linux:

* Python
  + Install: Follow instructions in Python documentation at, “[Installing Python 3 on Linux — The Hitchhiker's Guide to Python (python-guide.org)](https://docs.python-guide.org/starting/install3/linux/)”
* MongoDB
  + Install: Follow instructions in MongoDB documentation at, <https://www.mongodb.com/docs/manual/administration/install-on-linux/>
* Jupyter Notebook
  + Install: Follow instructions in Jupyter documentation at, <https://jupyter.org/install>

In Windows OS:

* Python
  + Install: Download the version and preferred installation package then follow the instructions in the deployment software, <https://www.python.org/downloads/windows/>
* MongoDB
  + Install: Follow instructions in MongoDB documentation at, <https://www.mongodb.com/docs/manual/tutorial/install-mongodb-on-windows/>
* Jupyter Notebook
  + Install: Download the Anaconda software package at <https://www.anaconda.com/download>, then download Jupyter Notebook through on Anaconda software package’s dashboard.

In MacOS:

* Python
  + Install: Download the version and preferred installation package then follow the instructions in the deployment software, <https://www.python.org/downloads/macos/>
* MongoDB
  + Install: Follow instructions in MongoDB documentation at, <https://www.mongodb.com/docs/manual/tutorial/install-mongodb-on-os-x/>
* Jupyter Notebook
  + Install: Follow the Anaconda software package documentation at <https://docs.anaconda.com/free/anaconda/install/mac-os/> , then download Jupyter Notebook through on Anaconda software package’s dashboard.

## Usage

### Code Example

Ex #1) 

Code Example #1 demonstrates the call that embeds the [www.snhu.edu](http://www.snhu.edu) link into the Grazioso Salvare logo image while also setting the image to 25% it’s original width and height within the html page.

Ex #2) A screen shot of a computer code

Description automatically generated

Code Example #2 demonstrates the call that sets the data table to Disaster Rescue or Inidividual Tracking training candidates following the process of reading the AAC animal database for all candidates matching the training requirements then setting the mongo cursor object to the records returned by the read operation.

Ex #3) 

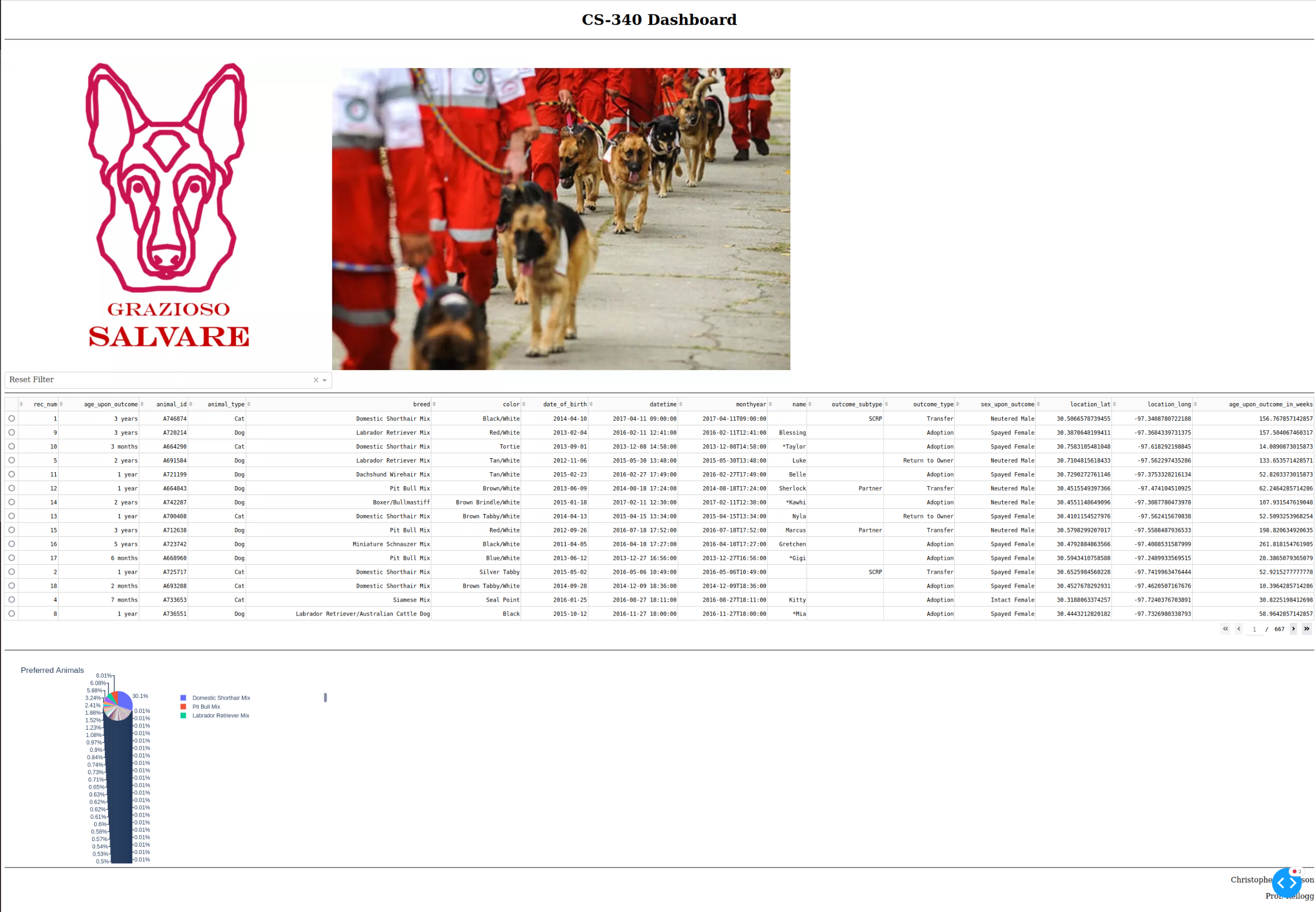
Code Example #3 demonstrates the callback routine of the update\_graphs function for receiving data updates from the object labelled datatable-id to allow the graph figure to match data within the datatable. The pie graph is then returned to an object labelled graph-id that is a child object of an separate Div, or namespace, that will display the current graph within the html page.

### Tests

This software package includes several tests to ensure that major faults are avoided. To test code in example one simply execute the software and click on the Grazioso Salvare Logo image to open the link to [www.snhu.edu](http://www.snhu.edu) and to test code in examples two as well as three a user will launch the software then click on the drop down menu, selecting the Disaster Rescue or Individual Tracking filter and observe that the data present within the graph and data table match the requirements of the database read statement. Additionally, this package includes test statements for faults within the drop down menu, errors in data passed to the graph/map constructors, and references faults within the operations of the CRUD python module for parameter faults along with database access faults.

### Screenshots

Initial Start-up:



Filter Selection (Mountain or Wilderness Rescue):

A group of people walking with dogs

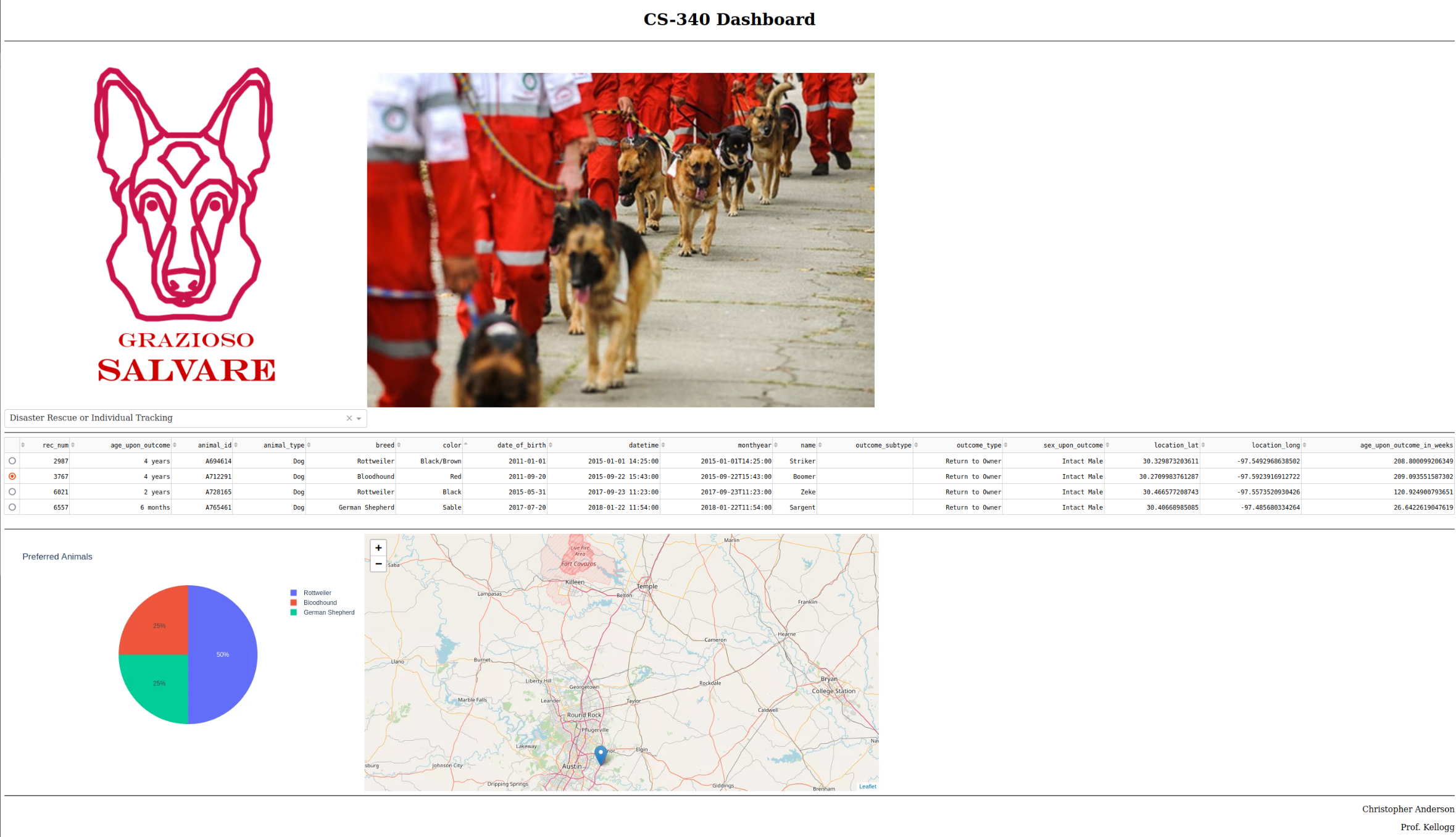
Description automatically generated

Candidate Selection (activates geolocation map):

A screenshot of a screenshot of a website

Description automatically generated

Column Sorting Options (candidates sorted by date of birth):



## Roadmap/Features

This software package provides Grazioso Salvare with the requested functionality by aligning the efficiency of the MongoDB database with the intuitive user-friendly interfaces provided by the Python libraries. The Dashboard module relies upon the Dash framework that acts by providing developers with a variety of template functions and objects for writing HTML webpages through a Python environment. Throughout the process of developing this application there were a variety of complications, primary stemming from interactions between the different software architypes since Python-to-MongoDB operations are unique in that they blend MongoDB’s language with Python and a developer must have a basic understanding of the HTML language to infer how Dash library functions will generate features of an application. When overcoming these complications, I referred to several HTML, MongoDB, and Dash framework reference websites to harden my understanding of the interactions between vital components. To complete this project, I first ensured that data was pasted from the MongoDB database to the data table, then I implemented the filter for rescue dog types and ensured that the data table updated properly. Next, I worked to have the data table reference the filtered data to the pie chart to ensure a synchronized environment.

## Contact

Your name: Christopher Anderson